REN-ISAC SECURITY ADVISORY

Logjam Vulnerability, and Hardening Your Cryptography

May 28, 2015

To: IT Executives and Security Staff

EXECUTIVE SUMMARY

Priorities:
MEDIUM, concerning response to the LOGJAM vulnerability
MEDIUM, concerning the more GENERAL CASE of hardening all institutional cryptography

Cryptographic protocols are used throughout your institution for the protection of data in transit, for example: user connections to secure web services, e-mail client-to-server and server-to-server connections, application server to database server connections, file transfers, and VPN.

The Logjam[1] cryptographic vulnerability was recently disclosed; and, there have been numerous previous cryptography-related vulnerabilities: notably BEAST[2], the recently highly-publicized Heartbleed[3] and POODLE[4], the BERserk vulnerability[5], and FREAK[6].

RECOMMENDATION: In response to recurring troubling vulnerabilities in cryptographic protocols we recommend that all sites:

1. Regularly identify their servers that use cryptographic algorithms for secure communications.
2. Assess the status of each of those servers.
3. Update server cryptographic libraries.
4. Harden server crypto configurations.

Response may be two-phased depending on your own local risk analysis: (1) address the immediate concern of configurations that are vulnerable to low-hanging fruit attacks, and (2) harden all institutional crypto.

Specifically concerning LOGJAM:

Affected services are those relying on HTTPS, SSH, IPsec, SMTPS, and protocols that rely on SSL/TLS.

Unlike Heartbleed, which permits sensitive information disclosure simply by probing a server in a crafted manner, a Logjam attack relies on Man-in-the-Middle (MITM) access, requiring an entirely separate form of attack to first establish that footing. That raises the complexity of a successful attack, however MITM should not be discounted as unlikely, too difficult to achieve, or something that "couldn't happen here".

MITM can be achieved via spoofed WiFi access points; and in this case, the VPN you rely on to protect users operating in public WiFi spaces may itself be vulnerable to the Logjam attack. Other MITM techniques such as ARP spoofing, rogue DHCP service, exploit of Web Cache Control Protocol, Web Proxy Auto Discovery, DNS poisoning, BGP route injection, and physical inline interception present varying levels of opportunity. High-value targets are worth the extra effort to an attacker; and/or a sophisticated attacker may have MITM capability already in place waiting for the opportunity something like Logjam presents.

---

[1] https://weakdh.org/
[2] https://en.wikipedia.org/wiki/Transport_Layer_Security#BEAST_attack
[3] http://www.ren-isac.net/alerts/REN-ISAC_Alert_HeartBleed_20140410.pdf
[4] http://www.ren-isac.net/alerts/REN-ISAC_Alert_POODLE_and_Crypto_20141022.pdf
[5] http://www.intelsecurity.com/advanced-threat-research/berserk.html
[6] https://freakattack.com/

Logjam allows a MITM attacker, at session establishment, to downgrade the session cryptographic protocol of vulnerable TLS connections to easily-broken ciphers. This allows the attacker to eavesdrop or modify data passing through the connection.

There are low-hanging fruit and difficult-to-achieve variations of the protocol downgrade vulnerability. These are based on the strengths of ciphers that your particular configuration supports. For example, if your configuration supports "export grade" ciphers (as many deployed configurations do, even though they shouldn't), a Logjam attack could downgrade the intended Diffie Hellman (DH) 1024 cipher to an easily broken DH 512 cipher.

The "next level up" in difficulty of an attack involves the use of DH 1024. Although DH 1024 is relatively secure, it's not immune to attacks by actors with nation state-like capabilities AND will become less secure over time.

RECOMMENDATION: We recommend that sites immediately disable support for export-grade ciphers and implement server configurations to use DH 2048.

The recommendation to eliminate export-grade ciphers (less than DH 1024) may be more easily achieved than implementing DH 2048 depending on your local environments and versions of server applications. For example, back-level but still supported versions of Apache (pre 2.4.7) don't permit configuration of a 2048-bit DH group[7]. Practical realities of back-level OS and systems may advise addressing the elimination of export-grade ciphers now and deferring DH 2048 until system upgrades can be accomplished; however, we recommend intentional and timely movement to DH 2048. Security should not be held captive to back-level systems.

Recommendations for mitigating the Logjam vulnerability are:

Reprinted with permission of WeakDH.org

What should I do?

If you run a server…

If you have a web or mail server, you should disable support for export cipher suites and generate a unique 2048-bit Diffie-Hellman group. We have published a Guide to Deploying Diffie-Hellman for TLS [1] with step-by-step instructions. If you use SSH, you should upgrade both your server and client installations to the most recent version of OpenSSH, which prefers Elliptic-Curve Diffie-Hellman Key Exchange.

If you use a browser…

Make sure you have the most recent version of your browser installed, and check for updates frequently. Google Chrome (including Android Browser), Mozilla Firefox, Microsoft Internet Explorer, and Apple Safari are all deploying fixes for the Logjam attack.

If you're a sysadmin or developer …

Make sure any TLS libraries you use are up-to-date and that you reject Diffie-Hellman Groups smaller than 1024-bit.

[1] https://weakdh.org/sysadmin.html

------- END OF EXECUTIVE SUMMARY ------

[7] https://wiki.mozilla.org/Security/Server_Side_TLS#Apache

The GENERAL CASE of Hardening All Institutional Cryptography

Step 1. Identify ALL Servers at Your Site That Use SSL/TLS

You can't check and fix SSL/TLS on servers that you don't know exist. Thus, the first step is to ensure that you have an inventory of all campus servers using SSL/TLS. Here are some ways you may be able to identify these servers:

-- ASK: Ask campus system administrators to self-identify any servers or appliances that may be using SSL/TLS (note that while SSL/TLS is most commonly used to secure web servers, it may also be used to protect SMTP and POP/IMAP, and for other types of network traffic, too). Be sure to think about systems hosted off-site/in the "cloud" as well as systems connected via your own local network.

-- CHECK: If you centrally manage your certificates (rather than delegating certificate procurement to individual sysadmins or departments), check your certificate authority's management console to see a list of systems that have obtained certificates.[8]

-- SCAN: Use an active network scanning tool to probe for systems doing SSL/TLS. Although servers may offer SSL/TLS secured services on any port, scanning should initially focus on TCP ports 443, 8443 (secure web), 465, 587, 993 (secure mail), and 22 (SSH). Note at this point we're not scanning for particular vulnerabilities, we're just trying to understand the population of servers that may need attention. Sites with centrally managed certificates may also be able to use a certificate discovery tool if one is provided as part of the certificate authority's management console or as a standalone tool. Other tools include, but are not limited to:

    Nmap: http://nmap.org/nsedoc/scripts/ssl-enum-ciphers.html
    Nessus: http://www.tenable.com/pvs-plugins/8548

-- SNIFF: Your intrusion detection system (Snort,[9] Bro,[10] etc.), Netflow traffic, or other passive methods may also help you to identify campus servers that are using SSL/TLS.

Step 2. Review the SSL/TLS Status of Each of Those Servers

Once you've identified all servers using SSL/TLS at your site, you should assess the status of those SSL/TLS implementations. This can be a complex process to undertake manually. Fortunately, however, you can use automated testing tools to do these tests for you. One highly-regarded tool for testing web servers is the Qualys SSL Labs SSL tester: https://www.ssllabs.com/ssltest/.

The Qualys tester provides an option to suppress public display of your server's results by ticking the box under the Domain Name box before submitting. After the scan completes, you'll receive a summary grade (Figure 1), as well as a detailed report. If your server is vulnerable to the low-hanging fruit version of the Logjam vulnerability, i.e. cipher suites <1024 bits, including "export grade", that will be noted in the SSLLabs server test Cipher Suites section (Figure 2) by "INSECURE" cipher suite notations.

[8] Note that if you use wildcard certificates (e.g., certificates that cover all systems in a domain, or certificates that cover all systems within a subdomain), it may be hard to identify all systems using such a certificate.
[9] https://www.snort.org/
[10] https://www.bro.org/

QUALYS® SSL LABS

Home Projects Qualys.com Contact

You are here: Home > Projects > SSL Server Test > dh768.serverhello.com

SSL Report: dh768.serverhello.com (109.123.120.179)

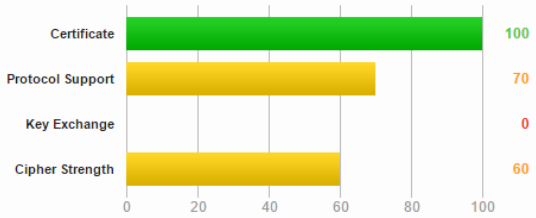
Assessed on: Wed, 20 May 2015 22:01:42 UTC | Clear cache

Scan Another »

Summary

Overall Rating

F



Visit our documentation page for more information, configuration guides, and books. Known issues are documented here.

This server supports insecure Diffie-Hellman (DH) key exchange parameters. Grade set to F. MORE INFO »

This site works only in browsers with SNI support.

This server supports TLS_FALLBACK_SCSV to prevent protocol downgrade attacks.

Figure 1

Configuration

Protocols

TLS 1.2	Yes
TLS 1.1	Yes
TLS 1.0	Yes
SSL 3	No
SSL 2	No

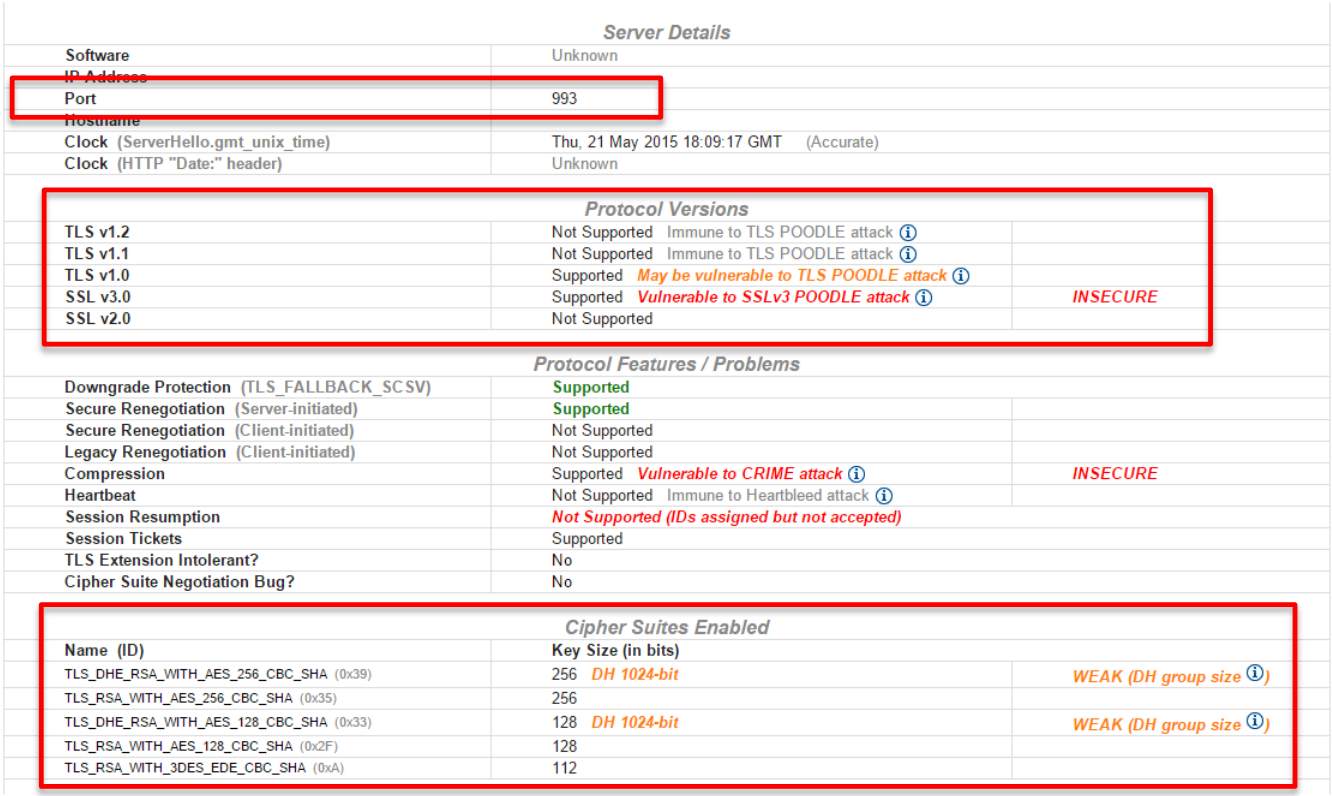
Cipher Suites (SSL 3+ suites in server-preferred order; deprecated and SSL 2 suites always at the end)

TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (0x9f)	DH 768 bits (p: 96, g: 1, Ys: 96) FS INSECURE	256
TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 (0x9e)	DH 768 bits (p: 96, g: 1, Ys: 96) FS INSECURE	128
TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 (0x6b)	DH 768 bits (p: 96, g: 1, Ys: 96) FS INSECURE	256
TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x39)	DH 768 bits (p: 96, g: 1, Ys: 96) FS INSECURE	256
TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 (0x67)	DH 768 bits (p: 96, g: 1, Ys: 96) FS INSECURE	128
TLS_DHE_RSA_WITH_AES_128_CBC_SHA (0x33)	DH 768 bits (p: 96, g: 1, Ys: 96) FS INSECURE	128
TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA (0x16)	DH 768 bits (p: 96, g: 1, Ys: 96) FS INSECURE	112

Figure 2

Q: "My server doesn't run on port 443. I can't figure out how to tell the Qualys tester to check some other port!"

A: If you have a server running SSL/TLS on some other port, you may need to use a different tool. For example, you may want to try the Comodo tester instead (see https://sslanalyzer.comodoca.com/). That tester will let you specify a hostname followed by a colon and a port number (e.g., serverfoo.example.edu:993). Partial output from that tester follows:



Server Details

Software	Unknown
IP Address	
Port	993
Hostname	
Clock (ServerHello.gmt_unix_time)	Thu, 21 May 2015 18:09:17 GMT (Accurate)
Clock (HTTP "Date:" header)	Unknown

Protocol Versions

TLS v1.2	Not Supported Immune to TLS POODLE attack	
TLS v1.1	Not Supported Immune to TLS POODLE attack	
TLS v1.0	Supported *May be vulnerable to TLS POODLE attack*	
SSL v3.0	Supported *Vulnerable to SSLv3 POODLE attack*	*INSECURE*
SSL v2.0	Not Supported	

Protocol Features / Problems

Downgrade Protection (TLS_FALLBACK_SCSV)	Supported	
Secure Renegotiation (Server-initiated)	Supported	
Secure Renegotiation (Client-initiated)	Not Supported	
Legacy Renegotiation (Client-initiated)	Not Supported	
Compression	Supported *Vulnerable to CRIME attack*	*INSECURE*
Heartbeat	Not Supported Immune to Heartbleed attack	
Session Resumption	*Not Supported (IDs assigned but not accepted)*	
Session Tickets	Supported	
TLS Extension Intolerant?	No	
Cipher Suite Negotiation Bug?	No	

Cipher Suites Enabled

Name (ID)	Key Size (in bits)	
TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x39)	256 *DH 1024-bit*	*WEAK (DH group size)*
TLS_RSA_WITH_AES_256_CBC_SHA (0x35)	256	
TLS_DHE_RSA_WITH_AES_128_CBC_SHA (0x33)	128 *DH 1024-bit*	*WEAK (DH group size)*
TLS_RSA_WITH_AES_128_CBC_SHA (0x2F)	128	
TLS_RSA_WITH_3DES_EDE_CBC_SHA (0xA)	112	

This example output shows that IMAP-SSL is poorly configured for protocol support (ref. Protocol Versions), but fortunately has protection from low-hanging fruit Logjam attacks by not supporting export grade ciphers.

IMPORTANT NOTE: Be sure to check the status of all SSL/TLS servers/services you identified in Step 1. https://www.[yourschooldomain].edu is not the only secure server at your school!

IMPORTANT NOTE: When updating your servers to fix Logjam, you should also fix any other security vulnerabilities identified during the testing process.

STEP 3. Update Your Servers' Cryptographic Libraries

Cryptographic libraries are continually being patched and enhanced. For example, OpenSSL, the most broadly used cryptographic library, was last patched on March 19 2015 to address twelve security vulnerabilities.[11] Two of the twelve were defined as HIGH severity. It's critical for cryptographic libraries to be patched up-to-date!

At the time this advisory was written, OpenSSL users should be running 1.0.2a, although further upgrades over time are certain to take place. See https://www.openssl.org/source/ for the most recently available version.

[11] https://www.openssl.org/news/secadv_20141015.txt

If openssl is in your path, you can check the version that's currently installed with the command:

```
$ openssl version
OpenSSL 1.0.1j 15 Oct 2014          ← Not up-to-date!
```

Important Note: After updating your cryptographic libraries, be sure to recompile and reinstall any applications that may be statically linked with the old cryptographic libraries.

Important Note: Ensure that any application you recompile with updated crypto libraries is itself fully up-to-date. For example, at the time this alert was written, the latest version of Apache was 2.4.12,[12] and the latest mainline version of nginx was 1.9.0.[13] After recompiling and reinstalling those applications, explicitly confirm that you're actually running the new version you've just built.[14]

Q. "I use an enterprise-grade commercially-supported Linux distribution that stresses stability, and it tends to lag when it comes to updates for OpenSSL and other applications that can leverage newer cryptographic and security features. Our local policy does not allow us to install ad-hoc packages except as distributed by the commercial Linux distribution vendor. What should I do?"

A. This is a difficult situation. If you are using a commercially-supported distribution, we encourage you to contact your vendor's support staff to see what they'd advise, and share your perspective on the importance of having current cryptographic libraries and security features even in distributions that strive to avoid unnecessary changes. Most vendors, even those that stress stability over everything else, understand the importance of incorporating critical security-related updates. Also note that some vendors will at times backport specific security fixes and enhancements into older versions of their distributions. Be sure to check the vendor's security notification announcements to determine if specific fixes have been backported.

Q. "We run Microsoft Windows Server -- what about our cryptographic libraries?"

A. You're using Schannel. Ensure you've applied all available service packs and patches. See also:

Microsoft IIS section of: https://weakdh.org/sysadmin.html

"Cipher Suites in Schannel"
http://msdn.microsoft.com/en-us/library/windows/desktop/aa374757%28v=vs.85%29.aspx

If you're using Microsoft IIS, you may find this free tool helpful:
https://www.nartac.com/Products/IISCrypto/Default.aspx

Step 4. Harden Your Server's Configuration

You're now ready to tweak your server's configuration to ensure it's doing crypto the way it should. While you're making that change, however, you should also be reviewing the totality of your server's SSL/TLS configuration.

---

[12] http://httpd.apache.org/download.cgi
[13] http://nginx.org/
[14] Some operating systems may have atypical conventions for the appropriate location of applications and other configuration files. If you build cryptographic libraries or applications from scratch AND you fail to adjust the default installation locations, you run the risk of installing updated cryptographic libraries or applications ALONGSIDE existing out-of-date applications. That is, rather than ending up with one current installation, you can easily end up with TWO parallel installations, one legacy installation and one updated installation. This can be very confusing. It is thus critical that you pay attention to the correct location for all libraries, header files, and binaries during configuration, build, and installation, and that you verify the version of the products you're running after you've concluded your upgrades.

If you are not a cryptography buff, you may wonder what settings you should be configuring -- there are a somewhat daunting set of options, and there are legitimate differences of opinion even among experts about how you should configure your systems, with the largest differences turning on the importance of compatibility (particularly for users with legacy systems) vs. strong security.

Examples of resources you may want to consult when it comes to configuring your server's crypto include:

Guide to Deploying Diffie-Hellman for TLS
https://weakdh.org/sysadmin.html

Mozilla Security/Server Side TLS
https://wiki.mozilla.org/Security/Server_Side_TLS

Applied Crypto Hardening
https://bettercrypto.org/static/applied-crypto-hardening.pdf

Cryptographic Best Practices in the Post-Snowden Era
http://www.educause.edu/sites/default/files/library/presentations/SEC14/SESS32/crypto-bcp.pdf

Qualys SSL/TLS Deployment Best Practices
https://www.ssllabs.com/downloads/SSL_TLS_Deployment_Best_Practices.pdf

Summary recommendations AS OF THIS TIME would include:

-- Immediately ensure that your servers DO NOT offer export-grade cipher protocols.

-- Implement DH 2048 ciphers.

-- Ensure that your users are running the most current versions of browsers.

-- Ensure your server's configuration DOES offers TLS 1.2. Ensure your server does NOT offer SSLv2, SSLv3, TLS 1.0, or TLS 1.1.

-- While you're tweaking your server, check to confirm that you **ARE using a SHA2 server certificate,** and NOT a SHA1 certificate. SHA1 certificates are relatively weak, and will begin to generate warnings in popular browsers (such as Chrome). If you are NOT using a SHA2 cert, request a SHA2 certificate from your certificate authority.

-- When you request that SHA2 certificate, consider generating a CSR with a **4096 bit RSA key**. Although a 2048 bit RSA key is the current industry standard/most commonly employed, most sites can do a stronger 4096 bit RSA key. Note that leveraging a 4096 bit key requires additional system resources for each connection. For sites with a moderate amount of traffic this should not be an issue. If you are hosting a high traffic site, and system resources are already near capacity, you should consult with system engineers to determine if the extra resources required by a 4096 bit key will exceed capacity.

-- Prioritize cipher suites that use **AES-256 or AES-128** as a symmetric cipher. Do NOT use weak ciphers with less than 128 bit keys, including so-called "export grade" ciphers. Do NOT use RC4. Ensure that server cipher preferences are honored.

-- Ensure you have prioritized cipher suites that use ephemeral key exchange. Ephemeral key exchange delivers **"forward secrecy,"** and protects any traffic that may have been vacuumed up and archived by an adversary, in the event your private keys are ever compromised. Having strong DHPARAMs is a key part of this. However note that some web servers do NOT allow you to specify the use of strong DHPARAMs.

-- Consider configuring your site to employ **http strict transport security.**[15]

-- If you are running a crypto library that supports **elliptic curve crypto** (ECC), consider evaluating it. ECC has the potential to offer a highly desirable combination of increased strength and enhanced performance.

Step 5. Retest Your Server(s)

After you've completed all tweaks and have restarted your newly reconfigured servers, retest them with the same tools mentioned in step 2, just to double check that the expected outcome was successful. Hopefully you'll now have secure systems/services earning top marks!

Step 6. Don't Forget About Web Browsers on Laptops/Workstations/Tablets/etc.

Now that your servers are in better shape, don't forget to ALSO check the browsers on laptops, workstations, tablets, smart phones, etc. Keeping them up-to-date is essential to your security. The process of checking browsers to make sure they're up-to-date will vary from browser to browser. For example, in Firefox, go to the Firefox menu item, and then go to "About Firefox" to automatically check for updates.

SSL/TLS capabilities of a browser can be tested at: https://www.ssllabs.com/ssltest/viewMyClient.html

And while you're working to ensure that your browsers are up to date, also check your systems for any updates that may be pending for *other* parts of your system.

MORE!

If you'd like to learn more about hardening your institutional cryptography we heartily recommend:
New Crypto 101, https://www.stsauver.com/joe/new-crypto-101/new-crypto-101.pdf, Joe St Sauver, Oct 2014

Feedback on this document?

We welcome your feedback on this document. Please send your comments or suggestions to soc@ren-isac.net

Credits

We wish to thank the REN-ISAC Technical Advisory Group[16] for assistance in developing this Alert and give particular thanks to Joe St Sauver of Farsight Security.

References

Copy of this Alert is available at:
http://www.ren-isac.net/alerts/REN-ISAC_Alert_Logjam_and_Crypto_20150528.pdf

About REN-ISAC: The REN-ISAC mission is to aid and promote cybersecurity operational protection and response within the research and higher education (R&E) communities. The mission is conducted through private information sharing within a community of trusted representatives at member organizations, and as a computer security incident response team (CSIRT) supporting the R&E community at-large. REN-ISAC serves as R&E's trusted partner in commercial, governmental and private information sharing relationships, in the formal U.S. ISAC community, and for served networks. http://www.ren-isac.net

[15] http://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security
[16] http://www.ren-isac.net/about/advisory.html